UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/730,975	12/10/2003	Efstratios Tsantilis	11884/406401	5120

23838 7590 07/10/2007

KENYON & KENYON LLP
1500 K STREET N.W.
SUITE 700
WASHINGTON, DC 20005

EXAMINER	
WANG, JUE S	

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
07/10/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/730,975	TSANTILIS, EFSTRATIOS	
	Examiner	Art Unit	
	Jue S. Wang	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *10 December 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-32* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-32* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *10 December 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-32 have been examined.

Specification

2. The specification is objected to because of the following informalities:

3. The title of the invention is not descriptive. The title does not mention determining backward compatibility based on a backward compatibility metric as recited in the claims. A new title is required that is clearly indicative of the invention to which the claims are directed.

Claim Rejections - 35 USC § 112

4. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

5. Claims 1-11, and 14-32 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

A. The following claim language is indefinite and not clear:

i. As per claim 1, line 2, and claim 25, lines 3-4, the phrase "assembling a baseline snapshot of a baseline version of a software interface" is used. This limitation is not clearly understood because it is not clear what the process of assembling entails (i.e., is the snapshot assembled by filtering out desired information from the software interface, or is the snapshot assembled by making a copy of the software interface?) and it is not clear what a snapshot contains (i.e.,

is the snapshot just a copy of the software interface taken at a particular time, or does the snapshot contain a selection of the information in the software interface, or does the snapshot contain metadata about the information in the software interface?).

ii. As per claim 1, line 11, claim 7, line 4, claim 28, line 6, the phrase "rating each detected difference according to a backward compatibility metric" is used. This limitation is not clearly understood because it is not clear how the rating works (i.e., do the ratings correspond to a level of risk of an updated snapshot being backward incompatible with the baseline version, or do the ratings correspond to the severity of the backward incompatibility?) and it is not clear how backward compatibility is measured by the metric (i.e., does the metric measure the number of sites where backward incompatibility occur, or does the metric measure the amount of effort required to fix the backward incompatibilities found?).

iii. As per claim 1, line 12-13, claim 7, lines 5-6, claim 25, lines 10-11, and claim 28, lines 7-8, the phrase "determining an overall backward compatibility ... based on the difference ratings" is used. This limitation is not clearly understood because it is not clear what is considered an "overall backward compatibility" (i.e., the percentage of code that is backward compatible, or just a yes/no answer to whether the code as a whole is considered backward compatible?) and how the "overall backward compatibility" is determined (i.e., is it determined by taking an

average of the ratings, or by comparing each of the ratings against some threshold?).

iv. As per claim 1, line 14, claim 7, line 7, claim 25, line 12, and claim 28, line 9, the phrase "issuing an alert containing the overall backward compatibility" is used. This limitation is not clearly understood because it is not clear why the alert is issued (i.e., is the alert issued when the overall backward compatibility indicates a backward incompatibility above a certain threshold or is the alert issued to report all backward incompatibilities found?).

v. As per claim 14, line 2, claim 28, line 3, and claim 32, line 8, the phrase "taking a first snapshot of a first software interface" is used. This limitation is not clearly understood because it is not clear what a snapshot contains (i.e., is the snapshot just a copy of the software interface taken at a particular time, or does the snapshot contain a selection of the information in the software interface, or does the snapshot contain metadata about the information in the software interface?) and it is not clear how the snapshot is taken (i.e., is the snapshot taken by filtering out desired information from the software interface, or is the snapshot taken by making a copy of the software interface?).

vi. As per claim 15, line 2 and claim 32, lines 14, the phrase "rating each detected difference according to a predetermined difference metric" is used. This limitation is not clearly understood because it is not clear how the rating works (i.e., do the ratings correspond to a level of risk of the second snapshot being backward incompatible with the first snapshot, or do the ratings correspond to a

degree of difference between the two snapshots?) and it is not clear what the predetermined difference metric measures (i.e., does the metric measure the number of sites where backward incompatibility occur, or does the metric measure the amount of effort required to fix potential bugs in the differences between the two versions?).

vii. As per claim 15, lines 3-4 and claim 32, line 16-17, the phrase "determining an overall difference ... based on the difference ratings" is used. This limitation is not clearly understood because it is not clear what is considered an "overall difference" (i.e., the percentage of code that is different, or just a yes/no answer to whether the code as a whole is considered different?) and how the "overall difference" is determined (i.e., is it determined by taking an average of the ratings, or by comparing each of the ratings against some threshold?).

Appropriate corrections are required.

Any claim not specifically addressed, above, is being rejected as incorporating the deficiencies of a claim upon which it depends.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this

subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

7. Claims 12 and 13 are rejected under 35 U.S.C. 102(e) as being anticipated by Schwab (US 6,986,132 B1).

8. As per claim 12, Schwab teaches the invention as claimed including a method for monitoring updates to a software object repository, comprising:

detecting at least one difference between a baseline version of an object interface and an updated version of the object interface (see Figs 20C-20D, column 24, line 35-column 26, line 15); and

issuing an alert when at least one of the detected differences indicates the updated version is not backward compatible (see column 25, line 23 – column 26, line 15).

9. As per claim 13, Schwab further teaches that the object interface comprises a list of components published by a software object residing in the object repository, said components including object properties and object methods (see Figs 11A, 18, Fig 20D, column 11, lines 51-57, and column 17, line 48 – column 18, line 5).

Claim Rejections - 35 USC § 103

10. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person

having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

11. Claims 14 and 25 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schwab (US 6,986,132 B1).

12. As per claim 14, Schwab teaches the invention as claimed including a method for comparing software interfaces, comprising:

detecting at least one difference between a first software interface and a second software interface (see Figs 20C-20D, column 24, line 35-column 26, line 15); and

issuing an alert when at least one of the detected differences indicates the first software interface is not backward compatible with respect to the second software interface (see column 25, line 23 – column 26, line 15).

Schwab does not explicitly teach taking a first snapshot of the first software interface and detecting at least one difference between the first and second snapshots. However, a snapshot is considered as a copy of the software interface made at a particular time, so it would have been obvious to one of ordinary skill in the art at the time of the invention that a copy of the API definition file can be easily made and that comparing two snapshots of two software interfaces would be equivalent to comparing the two software interfaces directly.

13. As per claim 25, Schwab teaches the invention as claimed including a computer programmed to monitor versions of a software interface, comprising:

means to store a baseline software interface (see Fig 20C);

means to compare an updated version of a software interface to the stored baseline software interface to detect at least one difference between the updated version and the baseline version (see Figs 20C-20D, column 24, line 35-column 26, line 15);

means to determine an overall backward compatibility of the updated version based on the detected differences (see column 25, line 23 – column 26, line 15); and

means to issue an alert message containing the overall backward compatibility (see column 25, line 23 – column 26, line 15).

Schwab does not explicitly teach means to assemble a baseline snapshot of a baseline version of a software interface, means to assemble an updated snapshot of an updated version of the software interface, and means to detect at least one difference between the two snapshots. However, a snapshot is considered as a copy of the software interface made at a particular time and the assembling process is considered as making a copy of the interface, so it would have been obvious that a snapshot of the API definition file can be easily assembled and that comparing two snapshots of two software interfaces would be equivalent to comparing the two software interfaces directly.

14. Claims 1-11, 15-24, 26-28, and 32 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schwab (US 6,986,132 B1) in view of Atallah et al. (US 7,069,474 B2, hereinafter Atallah).

15. As per claim 1, Schwab teaches a method for monitoring updates to a software repository, comprising:

obtaining a baseline version of a software interface over a network from a software repository via an application programming interface (see Fig 11A, 11C, Fig 20C, column 17, lines 36-67, column 19, lines 19-27, column 24, lines 24-32) ;

storing the baseline snapshot (see Fig 20C);

obtaining an updated version of the software interface over the network from the software repository via the application programming interface (see Fig 11A, 11C, Fig 20C, column 17, lines 36-67, column 19, lines 19-27, column 24, lines 24-32);

comparing the updated software interface to the baseline software interface to detect at least one difference between the updated version and the baseline version (see Figs 20C-20D, column 24, line 35-column 26, line 15); and

issuing an alert message containing an overall backward compatibility (see column 25, line 23 – column 26, line 15).

Schwab does not explicitly teach assembling a baseline snapshot of a baseline version of a software interface, assembling an updated snapshot of an updated version of the software interface, and comparing the two snapshots. However, a snapshot is considered as a copy of the software interface made at a particular time and the assembling process is considered as making a copy of the interface, so it would have been obvious that a snapshot of the API definition file can be easily assembled and that comparing two snapshots of two software interfaces would be equivalent to comparing the two software interfaces directly.

Schwab does not teach rating each detected difference according to a backward compatibility metric and determining an overall backward compatibility of the updated version based on the difference ratings.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including rating each detected difference according to a compatibility metric and determining an overall compatibility of the software products based on the difference ratings (see Figs 4A, 4B, abstract, lines 2-7, 11-14, column 5, line 64 - column 7, line 15).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwab to rate each detected difference according to a backward compatibility metric and determine an overall backward compatibility of the updated version based on the difference ratings as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow users to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

16. As per claim 2, Schwab teaches that the alert message is issued only when the overall backward compatibility indicates the updated version is not backward compatible (see column 25, line 23 – column 26, line 15).

17. As per claim 3, Schwab does not teach that the compatibility metric comprises a table of software modifications including backward-compatible software modifications and backward-incompatible modifications.

Atallah teaches maintaining tables used during the risk assessment process to determine the compatibility of binary files (see column 6, lines 22-28, 34-39, 46-50). In determining backward compatibility between API definition files, Schwab must have knowledge of what modifications preserve backward compatibility and what modifications break backward compatibility. It would have been obvious to one of ordinary skill in the art to maintain the backward compatible and backward incompatible modifications used in Schwab in a database or table as taught by Atallah to facilitate easy access and maintenance.

18. As per claim 4, Schwab further teaches that the backward incompatible software modifications include: deleting a parameter from a subroutine (see Fig 20D, column 10, lines 48-50 and column 24, lines 1-9); and deleting a field from a public data structure (see Fig 20D, column 25, lines 49-58).

19. As per claim 5, Schwab further teaches that the backward incompatible software modifications include: adding a mandatory parameter from a subroutine (see Fig 20D, column 10, lines 48-50 and column 24, lines 1-9; EN: adding a mandatory parameter to the method changes the parameters of the method); and adding a mandatory field to a public data structure (see Fig 20D, column 25, lines 49-58).

20. As per claim 6, Schwab further teaches that the backward-incompatible software modifications include: redefining an optional parameter as a mandatory parameter(see Fig 20D, column 10, lines 48-50 and column 24, lines 1-9); changing a parameter data type (see Fig 20D,

column 26, lines 6-10); and changing a public field data type (see Fig 20D, column 26, lines 1-10).

21. As per claim 7, Schwab teaches the invention as claimed including a method for monitoring software updates, comprising:

detecting at least one difference between a baseline version of a software object and an updated version of the software object (see Figs 20C-20D, column 24, line 35-column 26, line 15; EN: the backward compatibility between an earlier version and a revision binary files (software objects) are compared using their respective API definition files); and

issuing an alert message including an overall backward compatibility based on the difference detected (see column 25, line 23 – column 26, line 15).

Schwab does not teach rating each detected difference according to a backward compatibility metric and determining an overall backward compatibility of the updated version based on the difference ratings.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including rating each detected difference according to a compatibility metric and determining an overall compatibility of the software products based on the difference ratings (see Figs 4A, 4B, abstract, lines 2-7 and lines 11-14, column 5, line 64 - column 7, line 15).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwab to rate each detected difference according to a backward compatibility metric and determine an overall backward compatibility of the updated version based on the difference ratings as taught by Atallah because it is expensive and time consuming to purchase

new versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow user to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

22. As per claim 8, Schwab does not teach that the alert message includes a summary of each detected difference.

Atallah teaches that the alert message includes a summary of each detected difference (see column 7, lines 5-16).

23. As per claim 9, Schwab further teaches that the detecting comprises discovering that a parameter in the baseline version is missing from the updated version (see Fig 20D, column 10, lines 48-50 and column 26, lines 6-10; EN: changing the parameters to a method includes deleting one of the parameters).

24. As per claim 10, Schwab further teaches that the detecting comprises discovering that a parameter is optional in the baseline version, but the parameter is mandatory in the updated version (see Fig 20D, column 10, lines 48-50 and column 26, lines 6-10; EN: changing the parameters to a method includes changing a parameter from optional to mandatory).

25. As per claim 11, Schwab further teaches that the detecting comprises discovering that a parameter is defined as a different data type in the updated version (see Fig 20D, column 10, lines 48-50 and column 26, lines 6-10).

26. As per claim 15, Schwab does not teach rating each detected difference according to a predetermined difference metric; determining an overall difference between the first software interface and the second software interface based on the difference ratings; and incorporating the overall difference into the alert message.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including rating each detected difference according to a compatibility metric and determining an overall compatibility of the software products based on the difference ratings (see Figs 4A, 4B, abstract, lines 2-7 and lines 11-14, column 5, line 64 - column 7, line 15).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwab to rate each detected difference according to a predetermined metric and determine an overall difference between the first interface and the second interface based on the difference ratings as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow users to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

27. As per claim 16, Schwab does not teach assigning a user to the first snapshot; and issuing the alert message to the user.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including assigning a user to the application being assessed (see column 5, lines 40-52) and issuing the risk assessment report to the user (see column 7, lines 5-16).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwab to assign a user to the first snapshot; and issuing the alert message to the user as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow users to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

28. As per claim 17, Schwab does not teach incorporating a summary of each detected difference into the alert message.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including incorporating a summary of each detected compatibility difference into a report (see column 7, lines 5-16).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwab to incorporate a summary of each detected difference into the alert message as taught by Atallah because it is expensive and time consuming to purchase new

versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow user to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

29. As per claim 18, Schwab further teaches that the API definition file has at least one record, said record including at least one attribute (see Fig 20D, column 25, line 37 – column 26, line 10; En: attributes of classes, interfaces, fields and methods, are all considered as separate records in the API definition file) While Schwab does not teach organizing the records in a table, it would have been obvious that a table structure can be used since a table is a well known data structure in the art. Furthermore, it would have been obvious that a snapshot of the API definition file would have the records because the snapshot is considered as just a copy of the API definition file.

30. As per claim 19, Schwab further teaches that the record describes an object property (see Fig 20D, column 25, lines 49-58; EN: the fields of a class are considered as properties of the class since the fields of a class describe the class and class properties are considered object properties because it is well known in the art that an object is just an instance of a class).

31. As per claim 20, Schwab further teaches that the attribute indicates a data type of the object property (see Fig 20D, column 25, lines 49-58, column 26, lines 1-5; EN: the data type of the fields are considered data type of the object property).

32. As per claim 21, Schwab further teaches that the attribute indicates a data length of the object property (see Fig 20D, column 25, lines 49-58, column 26, lines 1-5; EN: length information is implied by the type information since a type of int requires a different amount of storage than a type of float).

33. As per claim 22, Schwab further teaches that the record describes an object method (see Fig 20D, item 1655, and column 26, lines 6-10).

34. As per claim 23, Schwab further teaches that the records describe an object method parameter (see Fig 20D, item 1655, and column 26, lines 6-10; EN: the object method parameter is described in the method signature).

35. As per claim 24, Schwab further teaches that the attribute indicates a data type of the object method parameter (see Fig 20D, item 1655, and column 26, lines 6-10; EN: the data type of the object method parameter is described in the method signature).

36. As per claim 26, Schwab does not teach that the computer has the means to rate each detected difference according to a backward compatibility metric.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including rating each detected difference according to a compatibility metric and determining an overall compatibility of the software products based on the difference ratings (see Figs 4A, 4B, abstract, lines 2-7 and lines 11-14, column 5, line 64 - column 7, line 15).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwab to rate each detected difference according to a backward compatibility metric as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow user to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

37. As per claim 27, Schwab does not teach that the computer has means to incorporate the ratings and the detected differences into the alert message.

Atallah teaches generating a report to indicate the binary compatibility risk profile with records to indicate failure, high risk, low risk, and guaranteed eligible (see Figs 4A, 4B, abstract, lines 10-16, column 5, line 64 - column 7, line 15).

38. As per claim 28, Schwab teaches the invention as claimed including a machine-readable medium having stored thereon a plurality of instructions for monitoring a software interface, the plurality of instructions comprising instructions to:

detect at least one difference between the first software interface and a second software interface (see Figs 20C-20D, column 24, line 35-column 26, line 15);

issue an alert message including an overall backward compatibility (see column 25, line 23 – column 26, line 15).

Schwab does not explicitly teach taking a first snapshot of the first software interface and detecting the differences between the first and second snapshots. However, a snapshot is considered as a copy of the software interface made at a particular time, so it would have been obvious to one of ordinary skill in the art at the time of the invention that a copy of the API definition file can be easily made and that comparing two snapshots of two software interfaces would be equivalent to comparing the two software interfaces directly.

Schwab does not teach rating each detected difference according to a backward compatibility metric; and determining an overall backward compatibility of the first software interface with respect to the second software interface, based on the difference ratings.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including rating each detected difference according to a compatibility metric and determining an overall compatibility of the software products based on the difference ratings (see Figs 4A, 4B, abstract, lines 2-7 and lines 11-14, column 5, line 64 - column 7, line 15).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwab to rate each detected difference according to a backward compatibility metric and determine an overall backward compatibility of the updated version based on the difference ratings as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating

systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow user to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

39. As per claim 32, this is the system claim of claim 28. Therefore, it is rejected using the same reasons as claim 28.

40. Claims 29-31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schwab (US 6,986,132 B1) in view of Atallah et al. (US 7,069,474 B2, hereinafter Atallah), as applied to claim 28 above, further in view of Theodossy et al. (US 6,898,768 B1, hereinafter Theodossy).

41. As per claim 29, Schwab and Atallah do not teach that the plurality of instructions stored on the machine-readable medium is executed in response to an external trigger.

Theodossy teaches a method for compatibility receiving a compatibility triggering event and verifying compatibility in response to the triggering event (see Fig 5, abstract, lines 4-7, column 3, lines 32-35, and column 6, lines 23-67).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwab as modified by Atallah such that the instructions are executed in response to an external trigger as taught by Theodossy because compatibility is critical for all upgrades and it is important to verify that compatibility exists for hardware revisions and software revisions prior to experiencing problems relating to incompatibility, and the triggering

event allows this verification to occur automatically (see column 2, lines 36-45 and column 3, lines 35-40 of Theodossy).

42. As per claim 30, Schwab and Atallah do not teach that the external trigger comprises a received notification of an update to the second software interface.

Theodossy teaches that the triggering event can be an attempt to perform a software upgrade on a system component (see column Fig 4, column 2, lines 48-55, column 5, line 52 – column 6, line 13, and column 6, line 23-63). While Theodossy does not specifically teach as such, it would have been obvious to one of ordinary skill in the art that the triggering event can be a notification of an update to the second software interface since an triggering event can be any event that necessitates the system to perform a compatibility verification to ensure optimal performance (see column 6, lines 29-33 of Theodossy) and update to a software interface would have an impact on other software interfaces relying on the updated software interface.

43. As per claim 31, Schwab and Atallah do not teach that the external trigger comprises a received notification of a scheduled event.

Theodossy teaches receiving a compatibility triggering event (see column Fig 4, column 2, lines 48-55, column 5, line 52 – column 6, line 13, and column 6, line 23-63). While Theodossy does not explicitly teach that the triggering event is a scheduled event, it would have been obvious to one of ordinary skill in the art that the triggering event could be scheduled since the triggering event can be upgrades on a system component and system upgrades can be a complex process requiring much planning (see column 1, lines 12-13).

Conclusion

44. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

- Apte (US 6,298,353 B1) is cited to teach checking serializations compatibility between versions of Java classes.
- Carter et al. (US 6,519,767 B1) is cited to teach a compiler and method for automatically building version compatible object applications.
- Strysniewicz et al. (US 6,591,417 B1) is cited to teach a method of and system for testing compatibility with an external API upgrade.
- Spring (US 6,971,093 B1) is cited to teach techniques for maintaining compatibility of a software core module and an interacting module.
- Perks et al. (US 7,191,196 B2) is cited to teach a method and system for maintaining forward and backward compatibility in flattened object streams.
- Lau et al. (US 7,191,435 B2) is cited to teach method and system for optimizing software upgrades.
- Das et al. (US 7,216,343 B2) is cited to teach a method and apparatus for automatic updating and testing of software.
- Ramachandran et al. (US 2005/0022176 A1) is cited to teach a method and apparatus for monitoring compatibility of software combinations.
- Runte et al. (US 2005/0086642 A1) is cited to teach tools for providing backwards compatible software.

45. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jue S. Wang whose telephone number is (571) 270-1655. The examiner can normally be reached on M-Th 7:30 am - 5:00pm (EST).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on 571-272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

J.W.
6/26/2007

MENG-AL T. AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100